Content-Based Group-Of-Picture Size Control in Distributed Video Coding

Content-Based Group-of-Picture Size Control in Distributed Video Coding

Enrico Masala, Yanmei Yu*, Xiaohai He

Abstract—Controlling the group of picture (GOP) size in distributed video coding (DVC) is a difficult but important task since it has a direct impact on the coding performance. This paper presents a framework to adaptively control the size of GOPs in a Wyner-Ziv encoder by means of encoder-side decisions based on support vector machines (SVM) that uses simple features extracted from the original video content. To train the SVM, firstly this work proposes how to compute the sequence of GOP sizes with the best rate-distortion performance given the set of GOP sizes that can be used during the encoding process. Then, an algorithm based on the previously trained SVMs is presented to control the actual GOP size each time a new decision can be taken at the encoder. Results show that the proposed algorithm can achieve a rate distortion performance close to the ideal one. Moreover, comparisons with a reference adaptive GOP size selection algorithm in the literature shows gains up to 2 dB PSNR in the best conditions.

Keywords: Distributed Video Coding, Adaptive GOP Size, Content-Based Adaptation, Support Vector Machine

1. Introduction

Distributed video coding (DVC) is a video compression paradigm different from traditional video compression standards such as H.264/AVC. Traditional video compression standards, usually with a complex encoder and a low-complexity decoder, are very well suited for applications such as broadcasting. However, for other applications such as compression and transmission of video from wireless surveillance cameras to a central server — a scenario characterized by low power, scarce memory and limited computational resources — other approaches may be more beneficial. In particular, DVC has been proposed for these types of applications. DVC is based on the Slepian-Wolf (SW) [1] and Wyner-Ziv (WZ) [2] theorems, which show that data could be

This work was supported in part by the EU FP7-PEOPLE-IRSES-2009 S2EuNet Project (grant no. 247083), National Natural Science Foundation of China under grant 61201388,and Young Teacher's Fund of Sichuan University under grant 2010SCU11007.

*The corresponding author is Yanmei Yu.

E. Masala is with the Control and Computer Engineering Department, Politecnico di Torino, corso Duca degli Abruzzi 24, 10129 Torino, Italy (e-mail: masala@polito.it, Tel: +390110907036, Fax: +390110907099).

Y. Yu and X. He are with the College of Electronics and Information Engineering, Sichuan University, No.24 South Section 1,Yihuan Road,610065 Chengdu, China (e-mail: yuyanmei@scu.edu.cn; hxh@scu.edu.cn, Tel: +86-28-85462766, Fax: +86-28-85463466).

compressed efficiently by independent encoders in lossless coding and lossy coding modes, respectively. The DVC approach reverses the complexity of the encoder and the decoder in traditional video coding standards by shifting the complex tasks to the decoder. Two early DVC practical architectures, i.e. the Berkeley PRISM (Power-efficient, Robust, hIgh compression Syndrome based Multimedia coding) Architecture [3,4] and the Stanford Architecture [5,6], were proposed independently in 2002. Among these two architectures, the Stanford architecture is the most popular one in the literature. This one firstly implemented Wyner-Ziv coding in the pixel domain [5], then it has been extended to the transform domain [7] to exploit the spatial statistical redundancy within a frame. In this paper, we focus on the Wyner-Ziv coding in the transform domain.

For the case of Wyner-Ziv coding, a number of technical issues need to be addressed to provide acceptable performance. For instance, the decision about whether to encode a frame as a key or a Wyner-Ziv (WZ) one is particularly important since it can have a large impact on the efficiency of the encoding process. To clarify the point, consider the case when the sequence can be easily predicted, due to, e.g., the presence of a large amount of static content. In this case, it may be more efficient to encode frames using more than one WZ frame between two key frames, thus making larger group of pictures (GOP). The opposite holds when the content is more difficult to predict from the key frames. Therefore, the encoder needs to decide, for each new frame, if the frame has to be compressed using traditional intra-coding techniques (i.e., as a key frame) or using the WZ approach. This activity must be performed using a low-complexity algorithm not to lose one of the main advantages of DVC, i.e., low encoder complexity.

In the case of traditional hybrid video codec scheme, motion vectors are available at the encoder and they can give good hints about how to optimally control the GOP size [8]. The DVC case, instead, requires to measure the activity present in the video content with low complexity techniques to run the algorithm at the encoder, as in [9][10][11], or analyze the video characteristics at the decoder side, where more complex algorithms could be used, and send the results back at the encoder by means of a feedback channel. A review of the state-of-the-art in the field is presented in Section 2.2.

This work builds on the DVC architecture by proposing a framework comprising two main parts: (1) the sequence analysis part which shows how to compute, with linear complexity in the number of frames, the ideal GOP structure for a video sequence which maximizes the rate-distortion performance given a set of available GOP sizes; (2) the use of the previous results on a set of sequences to train three SVMs that are the basic building blocks of the adaptive GOP size selection algorithm proposed in this work. For this aspect, a new content-adaptive frame type decision scheme is proposed which allows to select the best encoding mode by analyzing some features of the current and a few past frames. The frame type is decided by linear support vector machines (SVMs), which are trained with the sequence of GOP sizes that provides the best rate-distortion performance, as computed in (1). Although the proposed method is not — strictly speaking — a distributed algorithm, since the SVM works jointly on frames belonging to the key-frame group and the WZ group, its characteristics make the method suitable for the typical contexts in which DVC is used.

The main contributions of this work can be summarized as follows: 1) An algorithm to determine, with linear complexity in the number of frames, the best rate-distortion (RD) performance and the corresponding sequences of GOP sizes for a video sequence, given the results of a limited set of offline DVC encodings of the video sequence itself. 2) The definition of sets of features useful to determine the local characteristics of the video sequence. 3) An

algorithm to control the GOP size on the basis of the previous features. Simulation results based on many video sequences with different characteristics show the effectiveness of the proposed algorithm by means of comparisons with the ideal performance and with the technique proposed in [9].

The paper is organized as follows. Section 2 briefly reviews the DVC architecture used in this work, focusing on the GOP size issue. In Section 3 an algorithm is proposed to compute the best RD performance for any DVC sequence given a set of available GOP sizes and a number of offline DVC encodings of the video sequence itself. A proposal for a GOP size decision algorithm based on SVM is given in Section 4, followed by Section 5 which presents simulation results and comparisons with a reference technique. Conclusions are drawn in Section 6.

2. Background

2.1. DVC Coding Architecture

Several approaches have been proposed in the literature to implement DVC. In this work, we focus on the DISCOVER architecture [6][11][12]. In this framework, the encoder sends parity bits to the decoder in order to improve the quality of the side information (SI) available at the decoder. If the correlation between the side information and the current frame is high, less parity bits are needed to achieve a given quality, and vice versa. In order to generate the side information, the decoder uses interpolation mechanisms relying on the key frames which have been encoded using a traditional hybrid video coding approach, e.g., H.264/AVC.

As in the traditional hybrid video coding, frame interpolation is performed over groups of consecutive frames referred to as GOPs. However, the GOP size is typically allowed to vary during the encoding process to better exploit the temporal correlation, resulting in better performing temporal interpolation. In our method, for each new frame the encoder has to decide whether the frame has to be encoded as a key frame to start a new GOP, or as a WZ frame which increases the current GOP size.

WZ frames are typically more efficient in RD terms with respect to key frames if good side information estimation can be performed at the decoder. Static or easily predictable content is a typical situation where the WZ frames achieve high efficiency.

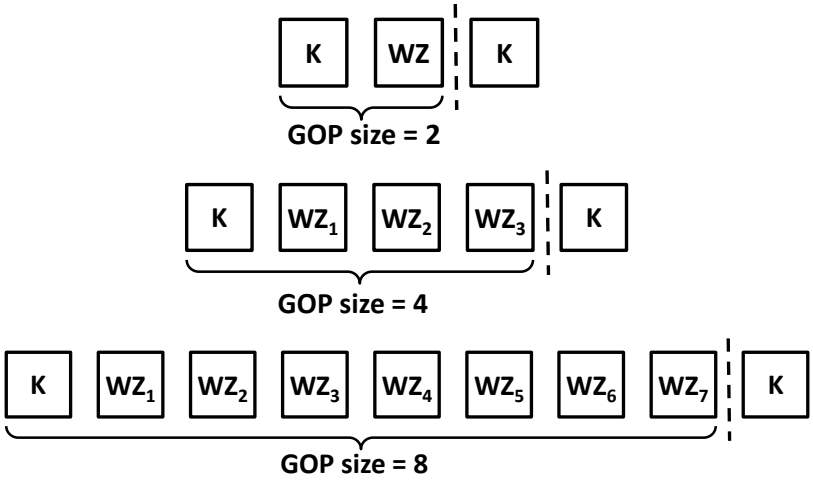


Fig. 1 Frame types for GOP size 2, 4, 8.

Determining the key and WZ frame position and frequency is very important to achieve a good RD performance. Consider a GOP size equal to two (Fig. 1): the intermediate WZ frame will be decoded using SI generated by means of the immediately previous and next key frames. If more WZ frames are present between key frames, the decoding process will need to perform several steps before reconstructing all WZ frames. For instance, for the case of GOP size 4 it may reconstruct an intermediate WZ frame, e.g., WZ_2, then it will proceed to decode the WZ_1 and WZ_3 frames.

Clearly, the characteristics of the video should be carefully considered in order to optimize the choice of the GOP size during the encoding process. Intuitively, when the amount of motion is high, the correlation among frames is low and smaller GOP sizes must be chosen. Conversely, if the amount of motion is low, correlation is higher and longer GOPs should be used, so to avoid the penalty of inserting a key frame when not needed, which typically accounts for a significant decrease of coding efficiency with respect to a well-interpolated WZ frame. However, note that selecting the most suitable GOP size is particularly difficult since the encoder has no access to the SI estimate done by the decoder.

2.2. Related Work

Until now most of the literature about the GOP size selection problem focused on algorithms at the encoder side, as in [9][10][11]. In [9] a number of frames of the input sequence equal to the maximum allowed GOP size is analyzed by computing some low-level features typically used for video parsing and indexing, such as difference of histograms, histogram of difference and variances, at the frame and block level, between adjacent frames. These features are assumed to contain information about the motion between adjacent frames. The features of each frame are arranged in a vector, each feature is normalized across the frames and an accumulated value is computed at each frame. Then, the minimum of the norm of the accumulated value is searched and the frames are grouped together. The procedure is repeated until all frames have been considered or the minimum of the norm is greater than a given threshold, i.e., it is not convenient to group the frames. The result of the procedure is a grouping of frames, which corresponds to GOPs that accumulated less motion, thus they are better correlated. The approach requires to store a number of frames, implying a certain amount of delay before a decision can be taken.

In [10] the authors propose a procedure to estimate the PSNR of the reconstructed WZ frames at encoder on the basis of entropy calculations and some partial reconstruction performed directly at the encoder. Also, in this case it is necessary to analyze a number of frames before deciding their encoding mode (key or WZ), thus a delay is introduced, which may not be compatible with the application requirements.

The work in [11] classifies blocks inside a frame as key, WZ or skip and depending on their relative amount it decides the frame type. Also, to enhance performance, a map is introduced to convey the information about the block type used during encoding. The thresholds needed for classification are obtained heuristically through experiments.

3. Computing the Ideal Rate-Distortion Upper Bound

To investigate the GOP size selection problem in more details, firstly we propose an algorithm that can determine the sequence of GOP sizes that maximizes the RD performance of a given

video sequence, thus determining the performance upper bound of an *ideal* adaptive GOP size technique. The algorithm relies on the fact that the rate and distortion of each GOP of a given size can be easily evaluated and then combined to determine the rate and distortion for a longer sequence by just summing up the values corresponding to each single GOP. The technique is ideal since in a practical transmission scenario it can not be employed at the encoder side because the rate and distortion values are not available there.

The problem is subdivided into two parts. First, the rate and distortion of any possible GOP in the sequence needs to be computed. Then, an algorithm will be presented to determine the sequence of GOP sizes that maximizes the rate-distortion performance.

3.1. Computation of the Rate-Distortion Values of Each GOP

Since the encoding of key frames is independent, computing the RD values for each GOP could be simply achieved by running the DVC encoder on subsequences of the original video that include the frames of interest, and then storing the rate and distortion values for those frames and GOP size. Given a sequence composed of L frames, it is possible to encode:

L GOPs of size 1, yielding L pairs of rate and distortion values, one for each GOP;

$L-1$ GOPs of size 2, yielding $L-1$ pairs of rate and distortion values, one for each GOP;

…

$L-(m-1)$ GOPs of size m, yielding $L-m$ pairs of rate and distortion values, one for each GOP.

The previous statements assume, of course, that the number of frames in the sequence is at least equal to m. Although running the encoding software each time to encode a single sub-video sequence would be possible, in practice to reduce the encoding runs that need to be performed the following procedure is exploited, without any loss of generality. GOPs with the same size are grouped together to make them consecutive, and shifted versions of the original sequence (i.e., where some initial frames have been dropped) are used to cover all the possible positions of the GOPs inside the original sequence. Fig. 2 shows an example of such a grouping. In the end, the result of running the encodings with the following scheme will produce a set of rate and distortion values for each frame that, combined together, allows to compute the rate and distortion pair of each GOP for any size and position in the video sequence.

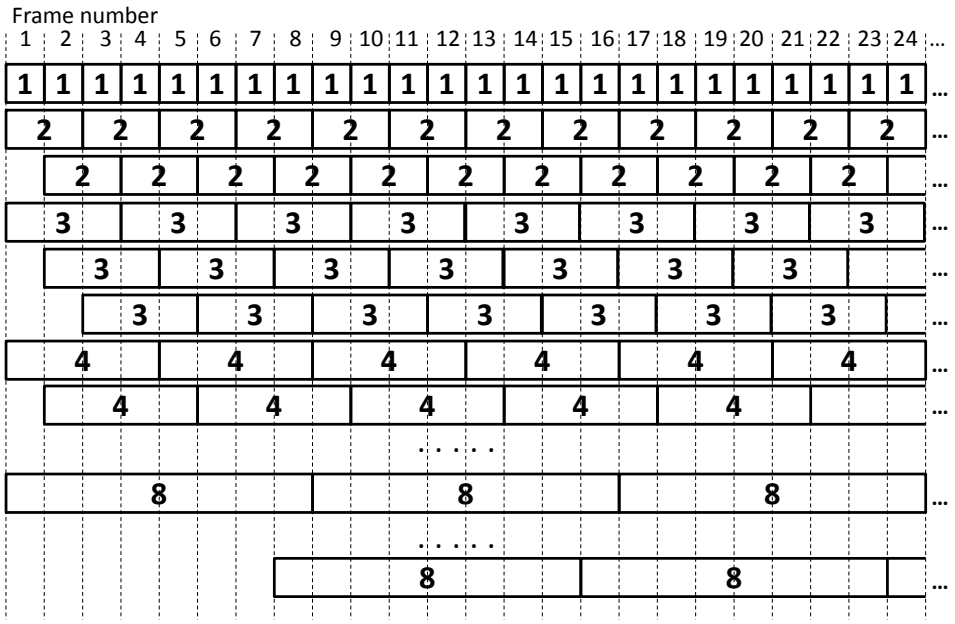


Fig. 2 Example of GOP grouping to reduce the number of runs of the encoder. The numbers in bold show the GOP size in frames.

Let *n* be the GOP size. For each GOP size, *n* different sequences are needed since with less sequences the GOPs would overlap and the number of runs of the encoder would not be sufficient. Thus, the number of runs of the encoder for such a scheme is $\sum_{n=1}^{m} n = m(m+1)/2$, assuming that the number of frames is at least *2m-1* because the larger GOP (of size *m*) starts at *m-1* frames from the beginning of the video sequence. By appropriately combining the rate *r* and distortion *d* of each frame from the encodings shown in Fig. 2 it is possible to compute all pairs $(r_{n,j}, d_{n,j})$ where *n* is the GOP size and *j* is the position of the start frame of the GOP in the sequence. Fig. 3 shows the *n,j* indexes superimposed on the GOPs shown in Fig. 2.

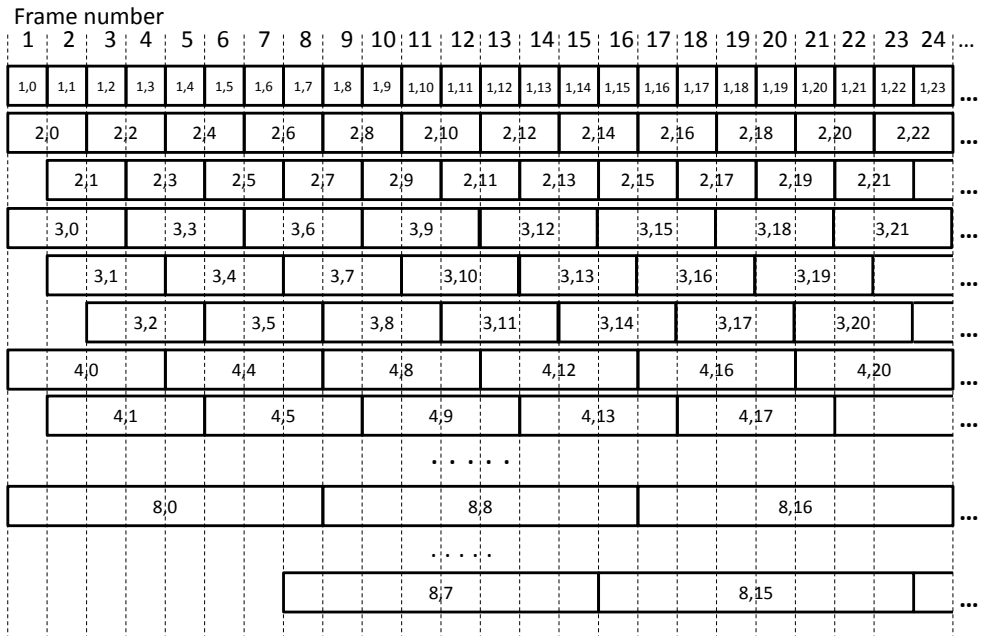


Fig. 3 Indexes of the GOPs in the form of *n,j* where *n* is the GOP size and *j* is the position in the video sequence (starting from zero).

The complete set of $(r_{n,j}, d_{n,j})$ allows to determine, without further runs of the encoder, the total rate and distortion for any arbitrary sequence of GOP sizes by simply summing the rate and distortion values of the consecutive GOPs taken at the correct position in the video sequence. For instance, for the case of consecutive GOPs whose size is 2, 8, 4, 2, 4, the total rate and distortion of the video sequence can be computed by summing the rate and distortion in the pairs $(r_{2,0}, d_{2,0}), (r_{8,2}, d_{8,2}), (r_{4,10}, d_{4,10}), (r_{2,14}, d_{2,14}), (r_{4,16}, d_{4,16})$. The previous arrangement ensures that the needed $(r_{n,j}, d_{n,j})$ value will be always available, since for each *n* all positions *j* are covered. Although any GOP size could be considered in principle by performing several encodings (36 in the case of maximum GOP size *m* equal to 8), during the development of the GOP size adaptation algorithm we will focus only on a subset of GOP sizes, i.e., 1, 2, 4, 8. In this case the number of encodings required to cover all possible combinations of GOP sizes and positions is 15, i.e., 1+2+4+8. This complexity reduction is possible due to the limited gain that the use of all GOP sizes, from 1 to 8, yields with respect to the case in which only GOP sizes 1, 2, 4, 8 are used. This is illustrated in Table I by means of the Bjontegaard difference (BD)[13] between the performance of the ideal GOP size adaptation strategy (described in more details in later sections) in the two cases. The average gain is very limited, 0.05 dB Peak Signal to Noise Ratio (PSNR). Further results and details are reported in the results section.

TABLE I
PERFORMANCE GAIN (BD-PSNR) WHEN ALL GOP SIZES ARE USED WITH RESPECT TO THE CASE OF GOP SIZE 1, 2, 4, 8 ONLY.

Sequence	Gain of GOP 1 to 8 w.r.t. GOP 1,2,4,8
Foreman	0.04
Hallmonitor	0.00
Mad	0.07
News	0.12
Paris	0.08
Students	0.04
Soccer	0.00

3.2. Computation of the Sequence of GOPs with the Best Rate-Distortion Performance

Although enumerating all possible combinations of GOP sizes that exactly match the number of frames *L* of the video sequence and computing their total rate and distortion values is possible in theory, the number of combinations grows exponentially with *L*. Thus, the following approach is used instead. Since we are interested in minimizing the distortion of the overall video sequence, we formulate the problem as:

$$\min_{S_i \in C} D(S_i) \quad \text{subject to} \quad R(S_i) \leq R_{\max} \tag{1}$$

where $S_i = (s_{i,1}, s_{i,2}, \cdots, s_{i,G_i})$ is a sequence of GOP sizes that exactly match the number of frames L of the video sequence (i.e., $\sum_{g=1}^{G_i} s_{i,g} = L$), C is the set containing all possible combinations, $D(S_i)$ and $R(S_i)$ are the total sequence distortion and rate. Each combination, identified by the index i, corresponds to the sequence of GOP sizes S_i and allows to easily compute $D(S_i)$ and $R(S_i)$ by means of the rate-distortion pairs previously computed. In particular, $D(S_i) = \sum_{g=1}^{G_i} d_{s_{i,g}, p_{i,g}}$ and $R(S_i) = \sum_{g=1}^{G_i} r_{s_{i,g}, p_{i,g}}$, where $p_{i,g}$ is the position of the first frame of the g-th GOP in combination i. More formally, $p_{i,1} = 0$ and $p_{i,g} = p_{i,g-1} + s_{i,g-1} = \sum_{k=1}^{g-1} s_{i,k} \; \forall g > 1$. Since r and d have already been computed for any possible GOP size and any valid position, evaluating the total rate and distortion of any arbitrary sequence of GOP sizes can be easily performed by summing all the relevant pre-computed values.

Referring to the problem in Eq. (1), note that the decision about using a given GOP size in combination i does not change the rate and distortion values of the other previous or subsequent GOPs, since they are isolated by the key frames, which in fact allows precomputing those values. The constrained minimization problem in Eq. (1) can be solved using a Lagrangian multiplier approach, i.e., the problem can be recasted as an unconstrained minimization of the form:

$$\min_{S_i \in C} J(S_i) = D(S_i) + \lambda R(S_i). \tag{2}$$

which can be easily solved as explained later in this section if λ is known. The λ value which matches the original R_{max} constraint can be determined by repeatedly solving the unconstrained minimization problem while, at each step, λ is updated until convergence using the bisection

algorithm in [14]. In terms of the standard Lagrangian formulation commonly found in rate-distortion optimization techniques [15], each GOP gives a different contribution to the total *J*, equal to $d + \lambda r$, where *d* and *r* are the total distortion and rate of the GOP. Note that we define $d = -q$, where *q* is the sum of the Peak Signal to Noise Ratio (PSNR) values, expressed in dB, for each frame in the GOP. Therefore, in the end the solution is exactly the one which maximizes the average PSNR of the encoded sequence.

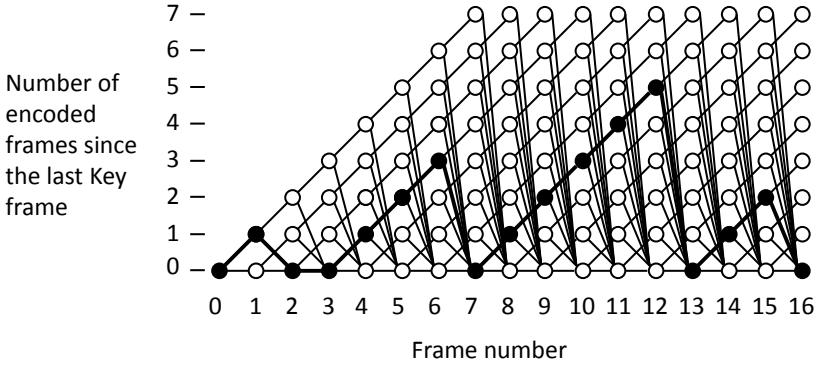


Fig. 4 Example of a trellis representing the possibilities for a 17-frame sequence. The black nodes show a sample path on the trellis corresponding to GOP sizes: 2, 1, 4, 6, 3.

Given a λ value, the unconstrained minimization problem in Eq. (2) can be solved by finding the minimum cost path on a trellis which represents the possible choices of the GOP size decision algorithm at each frame. An example of a generic trellis comprising all possible GOP size decisions from GOP size 1 up to GOP size 8 is shown in Fig. 4. The states (numbered from 0 to 7 on the y axis) correspond to the number of encoded frames since the last key frame. The transitions correspond to the decisions. The cost of each edge $C_f(x,x_{new})$ from state *x* at frame *f* to state x_{new} at frame *f+1* is detailed in the following. We denote the "number of encoded frames since the last key frame" as *x*. At frame *f*, in any state except where *x* is 7 two possible decisions are allowed: either delaying the insertion of a key frame (edge from state *x* to *x+1* at next frame, with cost $C_f(x,x+1)=0$) or inserting the key frame (edge from state *x* to 0, corresponding to the insertion of a GOP of size *x+1*, with edge cost $C_f(x,0) = d + \lambda r$ where *d* and *r* are the precomputed values for GOP size *x+1* ending at frame *f*). Finding the minimum cost path on that trellis is equivalent to finding the sequence of GOP sizes that minimizes *J* in Eq.(2). Clearly, the path must end at state 0 so that the GOP ends at the last frame in the encoded video sequence, i.e., the last frame is a key frame.

As already mentioned, it is possible to reduce the possible GOP size values without incurring in a significant performance degradation. In this work we focus on using GOP sizes 1, 2, 4, 8. Fig. 5 shows a modified trellis so that only these GOP sizes are allowed. In practice some states *x* (2, 4, 5, 6) now present mandatory transitions to state *x+1* at next frame since previous decisions do not allow to insert a key frame in that position. The minimum cost path computed on that trellis will result in the optimal solution of the problem in Eq.(2) when the possible GOP sizes are restricted to values 1, 2, 4, 8. The thicker edges in Fig. 5 shows a sample minimum cost path.

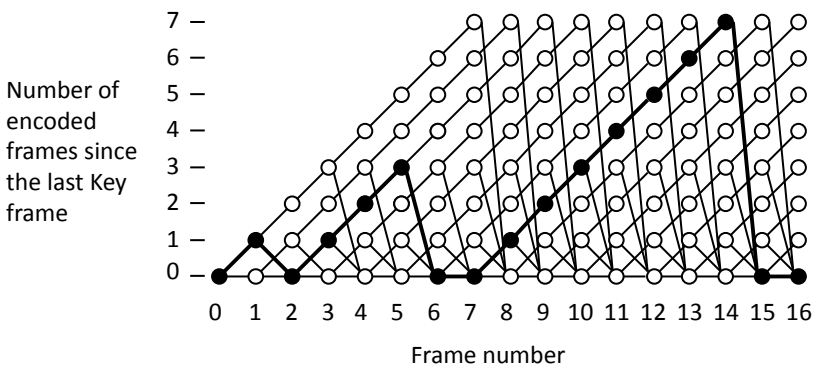


Fig. 5 Example of a trellis representing the possibilities for a 17-frame sequence when only GOP sizes 1, 2, 4, 8 are allowed. The black nodes show a sample path on the trellis corresponding to GOP sizes: 2, 4, 1, 8, 1.

The minimum cost path of the trellis can be easily found by means of the Viterbi algorithm that exhibits linear complexity with the number of frames *L* in the video sequence. The pseudocode of the algorithm is reported in the following.

For each x /* ranging from 0 to 7 included for GOP size 1 to 8 */
  $L(x)$ = 0; $S(x)$ = Empty list; /* Ordered sequence of numbers identifying the states */
For f = 1 to max_frame_num included /* e.g., max_frame_num=16 in Fig. 5 */
  For each state x_c at frame f with incoming edges
    /* find the incoming edge that leads to the minimum cost C for current node x_c */
    $x_m = \arg\min_x L(x) + C_{f-1}(x, x_c)$ /* cost may be 0 or $d + \lambda r$ as previously explained */
    $L'(x_c) = L(x_m) + C_{f-1}(x_m, x_{new})$; $S'(x_c) = S(x_m) + x_m$ /* append x_m to the list in S */
  $S = S'$; $L = L'$; /* move the data for each state x from new to old variables */
End: minimum cost path is in $S(0)$

At each frame, for each state it is necessary to determine which is the input edge that provides the minimum cost. For state 0 some comparisons are needed to find the minimum whereas for the other nodes no comparison is necessary to find the minimum since there is only one incoming edge, and they inherit the cost from the state at the previous frame to which they are connected due to the fact that the edge cost is 0. Also, in the most general case shown in Fig. 4 all states allow transitions to state 0, i.e., inserting a key frame. However, in the restricted case of GOP size 1, 2, 4, 8 shown in Fig. 5 only states 0, 1, 3 allow to choose the transition (node 7 presents a mandatory transition to node 0, i.e., the insertion of a key frame is mandatory since this is the maximum allowed GOP size).

The linear complexity with the number of frames of the Viterbi algorithm is a great advantage with respect to the impractical case in which all combinations have to be enumerated. However, this advantage stems from the fact that the original constrained minimization problem of Eq.(1) has been recasted into an unconstrained minimization using the Lagrangian multiplier. Although the Lagrangian multiplier approach is widely used in the literature to solve this type of problems, unfortunately it also introduces an approximation, i.e., among the points representing all the possible combinations of sequences of GOP sizes only those on the convex hull can be found by

the algorithm. An excellent explanation of the reasons related to this issue is reported in [15], p. 42. For the purpose of this paper it is sufficient to know that, once a λ value has been decided, running the Viterbi algorithm allows to find one of the points on the convex hull and the corresponding sequence of GOP sizes.

Only for the purpose of illustrating a sample convex hull, Fig. 6 shows the points corresponding to any possible sequence of GOP sizes when GOP sizes 1, 2, 4, 8 are considered for the first 22 frames of the video sequence *foreman*, as well as the convex hull itself. The points in the top part of the figure with increasing rate and PSNR are the ones that indeed maximize the PSNR as the rate constraint R_{max} is increased. However, only some of them, i.e., exactly the ones belonging to the convex hull, can be found by the Lagrangian approach for the reasons explained in [15]. The implementation of the Viterbi algorithm is very fast since it can find the solution, in terms of total rate, distortion and sequence of GOP sizes, for 300 different λ values, for a 300-frame sequence, on an Intel i5 CPU at 2.66 GHz in less than 2 seconds, i.e., one run for a given λ requires few milliseconds. Note that different λ values may yield the same result in terms of sequence of GOP sizes, i.e., they correspond to the same point on the convex hull. For instance, in the figure nine distinct results are found by testing 300 λ values. The nine points differ for the rate distortion tradeoff, which is determined by the λ value itself that yields that solution.

Note also that in a practical case there is no need to run the algorithm multiple times for different values of λ, since the λ value determines the local slope of the rate-distortion curve, i.e., the tradeoff between rate and distortion that is used to perform the encoding. While in principle different λ values could be used depending on the quantization parameter used by the DVC encoder, for simplicity in the results section we will keep this value fixed to represent a slope that approximates the average one of the DVC curves of the tested sequences. This point will be investigated further in future work. However, in practice, for the purpose of the results, the use of a single λ value instead of a set makes the corresponding performance a lower bound of what could be achieved by a technique that better adjusts the parameter.

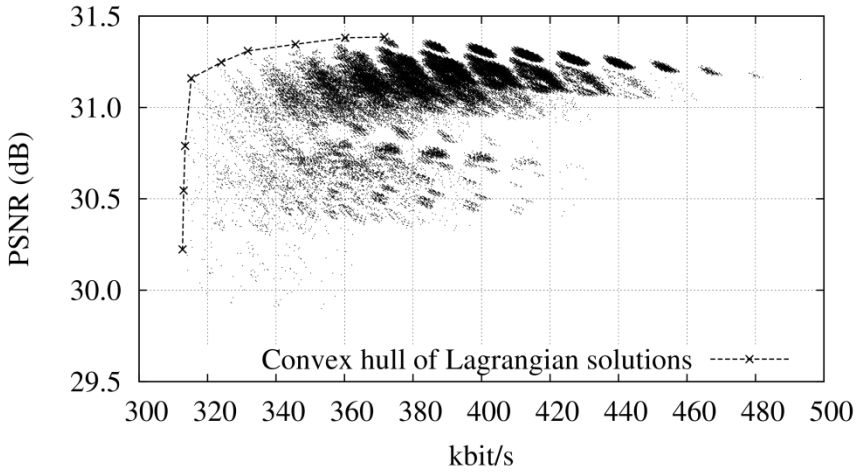


Fig. 6 Convex hull of the Lagrangian solutions, obtained by varying λ, for the first 22 frames of the *foreman* sequence (QP=39 for key frames and Q=1 for WZ frames). The dots corresponds to all possible combinations (90,600) of sequences of GOP sizes, considering GOP sizes 1,2,4,8.

4. Proposed GOP Size Selection Algorithm

The previous section showed that it is possible to compute the best solution of the problem of determining the sequence of GOP sizes that maximizes the performance, hence we have a solution that can be used in a learning algorithm to decide the GOP size every time a new decision can be taken. The algorithm will be trained for the best classification performance using the best GOP size values and it will decide the GOP size to be used at a certain frame during the encoding process as a function of a number of characteristics of the input video signal.

In more details, we rely on a discriminative model, i.e., a linear Support Vector Machine (SVM) [16]. To perform experiments with this type of SVM, we relied on the software described in [17][18]. For each event we try to detect, firstly the SVM is trained for the best classification performance with the values of a set of features (later described in this section) and the corresponding optimal outcome value, which is determined by means of the method described in Sec. 3. This process is performed once on a data set which is representative of a large variety of sequences. Then, the SVM is used to generate predictions that allow to decide the GOP sizes as explained in the following, and finally the performance is evaluated in terms of achieved video quality and bitrate. The previous section showed that at each frame, depending on the current state (i.e., the state in the trellis), when a decision has to be taken, only two possibilities are given, that is, close the GOP by inserting a key frame or leave it open. Hence we employed linear SVM decision machines that provide a binary outcome.

The first SVM, that we name S1, decides between two possibilities: either encoding a single-frame GOP or leaving the GOP open, i.e., encoding a two-, four- or eight-frame GOP. In fact, for some parts of sequences which are particularly difficult to compress using the DVC paradigm the single-frame GOP may be the most convenient choice. This typically happens in presence of strong motion inside the scene. If S1 decides not to use a one-frame GOP, a second SVM, named S2, decides between two possibilities: either encoding a two-frame GOP or leaving the GOP open, i.e., encoding a four- or eight-frame GOP. The process is repeated for a third SVM, named S4, that decides, at a later frame, between four- and eight-frame GOPs.

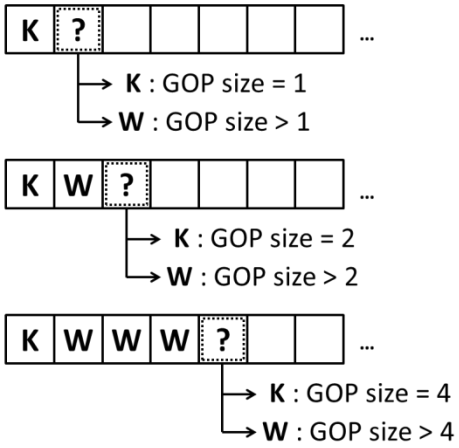


Fig. 7 Possible GOP size choices after one, two or four frames in the GOP have been encoded.

Differently from the case of traditional video coding based on the hybrid model, the GOP size decision can be taken at the end of the GOP, by simply inserting a key frame, rather than at the

beginning, since every frame is coded independently of the others. Fig. 7 illustrates the situation. Therefore, both S1 and S2 have the possibility to access the original video data of the whole GOP when they determine that either a one- or two-frame GOP has to be used. On the contrary, only the first three frames of the GOP are available when a four- or eight-frame GOP is chosen. S4 is used only at the fourth frame of the GOP, and it decides between a four-frame GOP, i.e., closing the GOP with a key frame and an eight-frame GOP, i.e., to continue with WZ frames until the eighth frame.

4.1. Features Employed by S1, S2 and S4

Meaningful features should be provided to the SVM in order to decide between the two possible output values. For this aim, we decided to construct features that resemble the interpolation approach used at the decoder to construct the SI but avoid the complexity due to the motion estimation. In fact, while more advanced features could be computed, e.g. using a fast motion estimation algorithm, our idea was to limit the complexity introduced at the encoder, thus refraining from using even simple motion estimation algorithms at the encoder side. The rationale behind this approach is to compute features that are an effective indication about how difficult it is to generate the side information from the available frames. The feature computation relies on a common basic procedure that we name *H( )*, that operates on a set of 8x8 integer values. The procedure counts the values that, taken without the sign, exceed a given threshold, set to 4 in our experiments. Thus the return value ranges from 0 to 64. The rationale behind this choice is that we want to identify values whose magnitude is significant (i.e., above a given threshold), and then equalize their contribution by simply counting them (instead of using their sum or similar metrics). The threshold value has been experimentally determined.

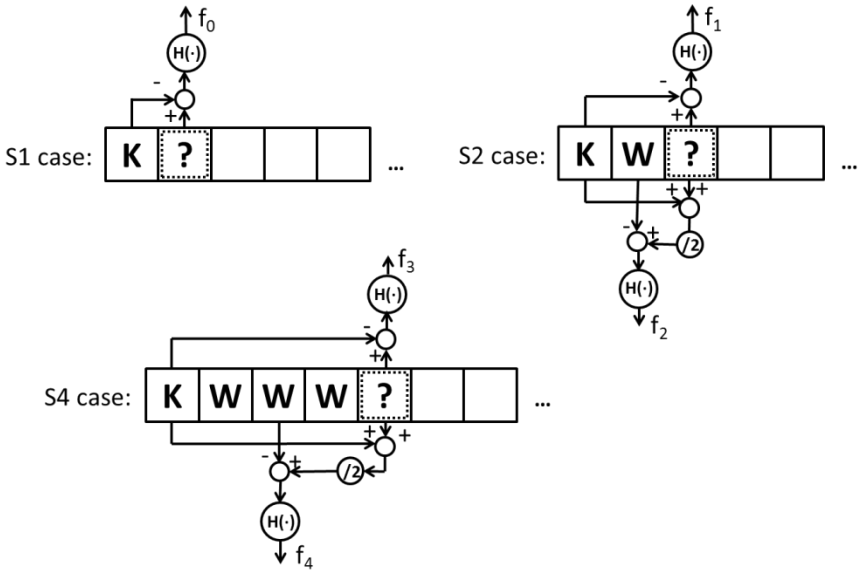


Fig. 8 Block diagram of the procedure to compute the input features for the S1, S2 and S4 case.

For S1 we employed one set of features. First, we consider the luminance component of the original uncompressed video sequence. The pixel-by-pixel difference between the current frame at position *p* and the frame at position *p-1* is subdivided into 8x8 blocks. Every block is passed to the *H( )* procedure, thus computing all the features in the first set of features named f_0.

For S2 we employed two sets of features. As in the previous case, we consider the luminance component of the original uncompressed video sequence. The pixel-by-pixel difference between the current frame at position *p* and the frame at position *p-2* is subdivided into 8x8 blocks. Every block is passed to the *H( )* procedure, thus computing all the features in the first set of features named f_1. To compute the second set of features (f_2), also the frame at position *p-1* is considered. The luminance component of frames at position *p* and *p-2* are averaged pixel-by-pixel, then the pixel-by-pixel difference between the average and the frame at position *p-1* is computed. As for f_1, the result, subdivided into 8x8 blocks, is passed to the *H( )* procedure to create the feature in the f_2 set. For the S4 case, the procedure is analogous but the position of the frames change. In particular, f_3 is computed applying *H( )* to the difference between the current frame at position *p* and the frame at position *p-4*, while f_4 applies *H( )* to the difference between the frame at position *p-2* and the average between the frames at position *p* and *p-4*. Fig. 8 provides a graphical illustration of the procedure.

The rationale behind this approach is to provide S2 with an indication about how difficult it is to generate the side information for the current frame on the basis of the previous key frame (f_1), as well as how difficult it is to generate the intermediate WZ frame if the current one is coded as a key one (f_2). These features help in deciding if a GOP with size equal to two should be used or not. A similar consideration hold for S4.

4.2. GOP Size Decision

The whole decision process is summarized by the flow chart in Fig. 9, which depicts the operations needed to encode a whole sequence. Once a new GOP is started, the first frame is coded as key, then S1 decides if there is strong motion, i.e., it is convenient to use another key frame, or if the current GOP size can be increased by inserting a WZ frame. In the latter case the current GOP size becomes two, therefore S2 has to decide, on the basis of f_1 and f_2, if a new GOP should be started (hence the frame is coded as a key one then the whole process is restarted) or the GOP should continue towards size equal to 4 or 8. When the current GOP size reaches 4, the S4 decides if the GOP size should be 4 (hence the frame is coded as key then the whole process is restarted) or the encoding process should proceed until the size equal to 8 is reached. At this point, in any case, a key frame is coded and the whole process is restarted.

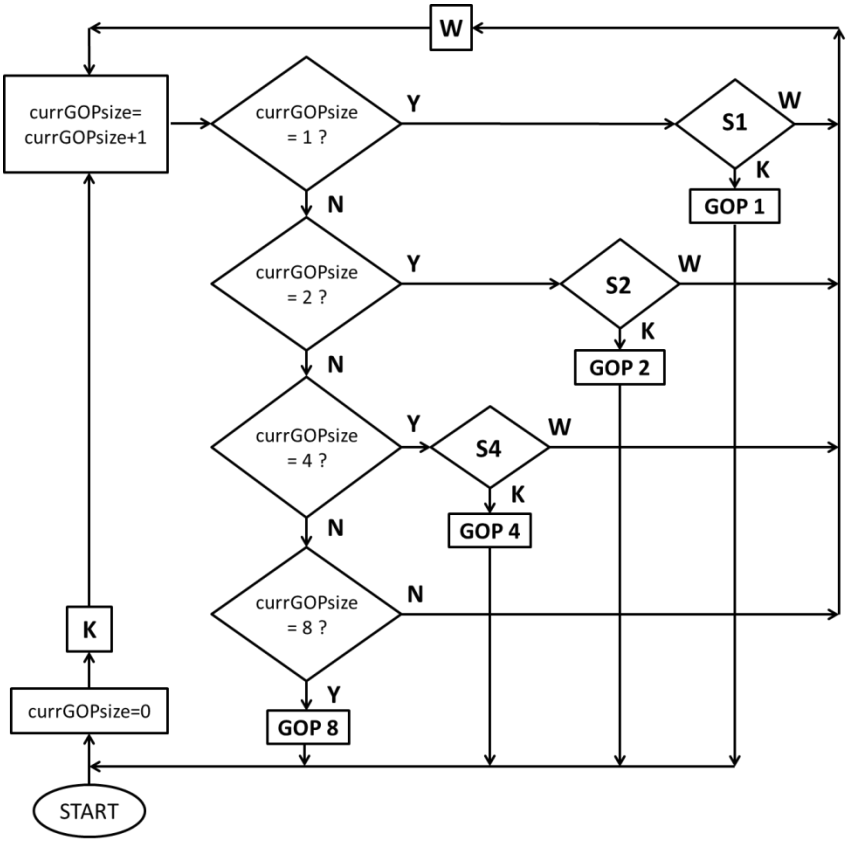


Fig. 9 Adaptive GOP size decision process. The *currGOPsize* variable indicates the current number of frames in the GOP. K and W represent the two possible decisions for each frame. For convenience, GOP 1, GOP 2, GOP 4 and GOP 8 indicate when the corresponding GOP is closed.

5. Results and Discussion

Sixteen sequences, listed in Table II, have been considered in order to achieve statistically significant results. All sequences have CIF resolution (352x288), 30 fps. The first 300 frames have been used in the experiments. Sequences were obtained from [19][20]. They present widely different characteristics in terms of video content. Some of the sequences have been used as training and some as the test set as shown in Table III. The large and widely different characteristics of the sequences in the training set allows to find good parameters for the SVMs as shown by the performance results.

Sequences have been encoded using the DISCOVER codec software freely available at [21] in executable format, which provides a transform-domain DVC system. The algorithms employed in this software are described in [11][22][23]. The software allows to set a so-called "Q" parameter for the WZ frames, which in practice is an index that selects a predetermined quantization matrix for WZ frame encoding. For the case of key frames, a standard H.264/AVC codec is used to encode intra frames. The quantization parameter (QP) of the key frames has been adjusted, by trial and error, for each pair of sequence and Q value of the DVC codec, so that both key and WZ frames have approximately the same average PSNR. The detailed set of QP parameters for the key frames corresponding to the various Q values of the WZ frames is reported in Table II. As

described in Section 3.2, for simplicity the λ value is the same throughout all the experiments. Its value has been chosen on the basis of the rate-distortion curves of the considered sequences, taking the two rate-distortion points corresponding to the coarser and finer quantization levels of each curve, computing the slope, then averaging the values. The resulting λ corresponds to a slope in the RD plane equal to 3.95/1000 (dB/ kbit/s). An algorithm to adjust the λ value for each sequence and quantization level could be used, however its design is difficult and left for future work. For the purpose of the results, the use of a single λ value makes the performance of the ideal GOP size adaptation technique suboptimal with respect to the one that could be achieved by better adjusting the λ parameter. However, despite this limitation, for all the presented results except some points of the *foreman* sequence the performance of the ideal GOP size adaptation technique is always better with respect to the one of other GOP adaptation strategies.

TABLE II
LIST OF VIDEO SEQUENCES

Name	QP values of key frames for WZ frames with Q=1,2,3,4,5,6,7,8
Akiyo	27, 26, 26, 25, 25, 24, 23, 20
Bowing	32, 30, 30, 28, 28, 26, 24, 20
Coastguard	37, 36, 36, 34, 34, 33, 30, 26
Container	27, 26, 26, 26, 25, 25, 24, 22
Foreman	39, 37, 37, 35, 34, 33, 31, 27
Football	43, 41, 41, 38, 37, 36, 32, 27
Hallmonitor	36, 35, 35, 33, 33, 31, 30, 26
Mad	28, 28, 28, 27, 26, 25, 24, 21
News	38, 37, 37, 34, 34, 33, 30, 26
Ntia diner	30, 29, 29, 28, 28, 27, 25, 22
Ntia spectrum1	32, 31, 31, 29, 29, 28, 26, 22
Ntia spectrum2	29, 28, 28, 27, 27, 26, 25, 22
Paris	37, 36, 36, 34, 33, 32, 30, 27
Sean	27, 26, 26, 26, 26, 25, 24, 22
Soccer	43, 41, 41, 37, 37, 35, 31, 26
Students	29, 28, 28, 27, 27, 26, 25, 22

TABLE III
TRAINING AND TESTING SETS

Training	Testing
Akiyo	Foreman
Bowing	Hallmonitor
Coastguard	Mad
Container	News
Football	Paris
Ntia diner	Students
Ntia spectrum1	Soccer
Ntia spectrum2	
Sean	

The results are presented in the form of a RD curve for Q values ranging from 1 to 8. First, comparisons are made to evaluate the performance difference due to the use of different sets of GOP sizes. We considered the following sets: GOP size 2,4,8; GOP size 1,2,4,8; GOP sizes from 1 to 8. Fig. 10 shows that for some sequences such as *foreman* the benefit of using GOP size 1 is significant, since strong motion is present in the video sequence. Conversely, the benefit of increasing the possible choices of GOP sizes from 1,2,4,8 to the whole 1 to 8 range is much more limited. For other sequences the benefit is even less, as in the case of *mad*.

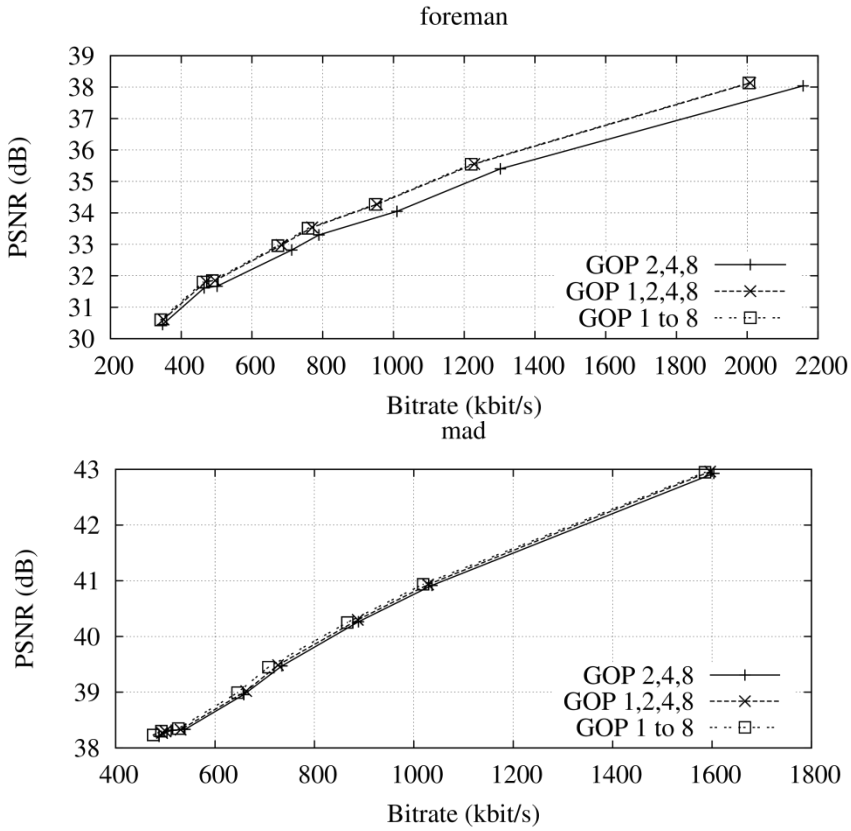


Fig. 10 Performance comparison of the ideal GOP size allocation strategy when different sets of GOP sizes are allowed.

To better quantify the difference between the various curves and to save space, the Bjontegaard difference (BD)[13] has been computed. This is a standard technique originally proposed in the context of video codec standardization that is gaining increasing diffusion in the multimedia research community. The basic idea is that given two sets of rate-distortion points, this technique fits a polynomial curve to the two sets and evaluates the average difference (in terms of PSNR or rate) by means of computing the integrals of the two fitted curves. Details are explained in [13]. Table IV shows the BD-PSNR values comparing the performance achieved by allowing all the GOP sizes (from 1 to 8) or only the values 1,2,4,8 to the case in which the GOP size is restricted to 2,4,8 for the tested sequences. Other results have already been presented in Table I with respect to the case of GOP size restricted to 1,2,4,8. The tables, especially Table I, quantitatively confirm the results for the *foreman* sequence and shows that only the *news* sequence can have some improvement, probably due to the fact that some small scene changes are present in the video sequence played in the background, which benefits from the use of GOP size 1.

To better visualize the difference in the GOP sizes used for the various video sequences by the ideal GOP size adaptation algorithm described in Section 3.2, Fig. 11 shows, in the form of a histogram, the number of frames used for each GOP size. For instance, for the *foreman* sequence, approximately 140 frames out of 300 belong to GOP size 2, thus the algorithm uses about 70 GOPs of size 2, in addition to GOPs of other sizes, to encode the video sequence.

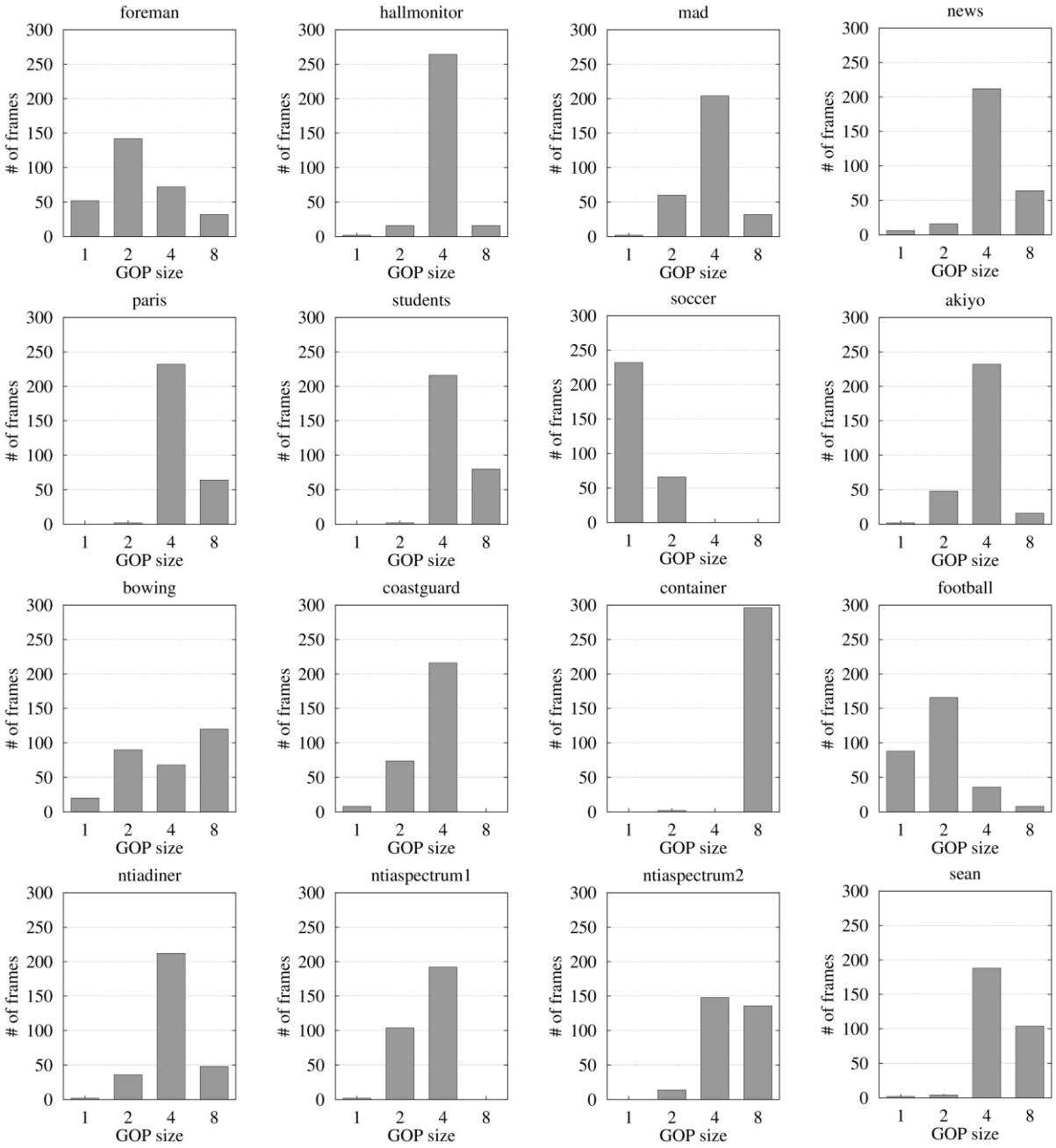


Fig. 11 Histogram of the number of frames using a given GOP size for the considered sequences for the 'ideal' GOP size adaptation algorithm described in Section 3.2.

TABLE IV
PERFORMANCE GAIN (BD-PSNR) WHEN DIFFERENT SETS OF GOP SIZES ARE USED

Sequence	Gain of GOP 1 to 8 w.r.t. GOP 2,4,8	Gain of GOP 1,2,4,8 w.r.t. GOP 2,4,8
Foreman	0.23	0.19
Hallmonitor	0.09	0.09
Mad	0.16	0.10
News	0.29	0.16
Paris	0.08	0.01
Students	0.04	0.00
Soccer	1.01	1.01

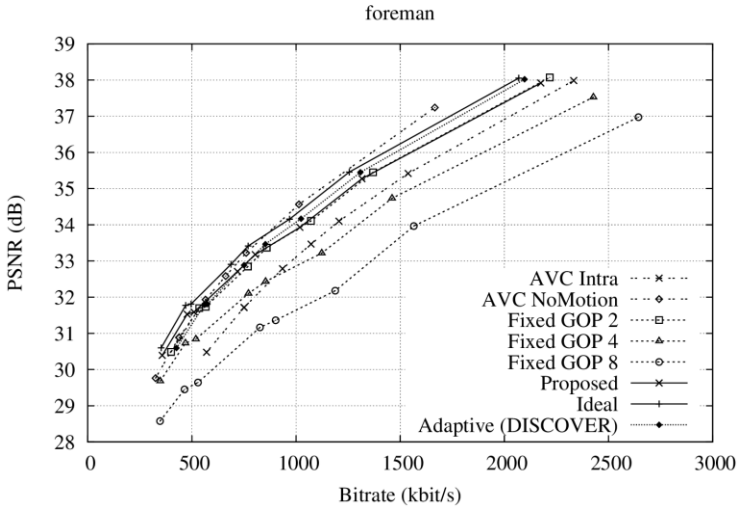
Another set of results compares the RD curve of the proposed algorithm to the adaptive GOP size decision algorithm included in the DISCOVER software [21], named "Adaptive (DISCOVER)" in the graphs. This algorithm is presented in details in [9] and summarized in Section 2.2. It is used as a reference term for the performance of the proposed algorithm. To better compare the two algorithms, all graphs also report the performance of the "ideal" sequence of GOP sizes as well as the performance of three fixed GOP size strategies (2, 4 or 8). The "ideal" performance has been computed with the algorithm proposed in Section 3.2, i.e., for each sequence and for each Q value the problem stated by Eq. (2) has been solved. In other words, once a λ value has been decided (the value used in the graphs is stated at the beginning of this section), the minimum cost path on the trellis has been found as described in Section 3.2 and the corresponding rate-PSNR pair has been computed and plotted in the graph. Finally, as a reference, the performance of the standard H.264/AVC codec using only intra frames as well as with the no-motion configuration using the I-B-I frame coding pattern are also shown in the graphs.

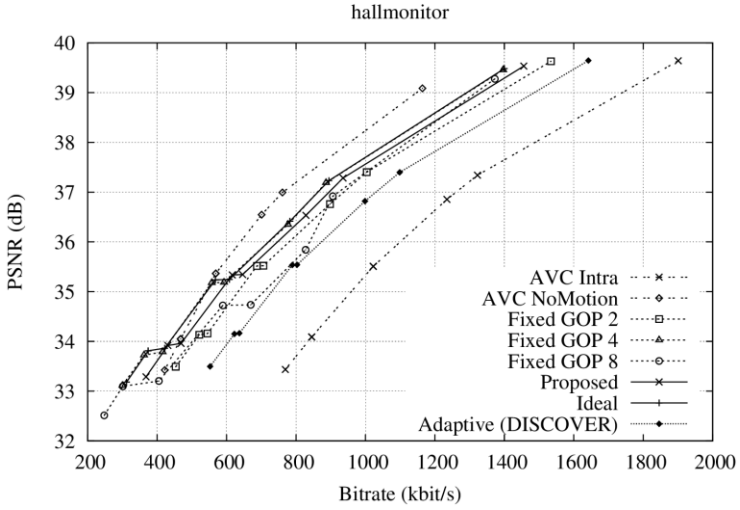
Results are shown in Fig. 12 for the tested sequences. In almost all cases the proposed technique approaches the performance of the ideal technique. Also, in many cases the result improves or it is very close to the performance achieved with a fixed GOP size strategy. Although the fixed GOP size strategy might appear trivial, note that the optimal size is not known in advance at the encoder, since it varies for each sequence according to the video content, and the characteristics of a real video may vary over time. In all cases, except some Q values for the *foreman* sequence, the performance is better than the one provided by the adaptive GOP size decision algorithm included in the DISCOVER software [21], despite this algorithm, differently from ours, can freely choose the GOP size among 1 and 8. In particular, GOP 1 is very beneficial for the *foreman* sequence, as it can be seen from observing the performance of the "AVC Intra" curve. For *foreman*, looking at the pattern generated by the adaptive DISCOVER algorithm we noticed that, in general, the algorithm inserts many GOP 1 in the encoding. Although this is not the optimal solution since the "ideal" curve still outperforms the adaptive DISCOVER one, the performance is good. Thus, on one hand, overestimating the number of GOP 1 encoding structures yields benefits for sequences with high motion such as *foreman*, but on the other hand the performance is much worse for other sequences as shown in the results graphs. Therefore, we believe that our proposed method, although it does not strictly outperform the "adaptive DISCOVER" for *foreman*, is still valuable since the gain for other sequences is significant.

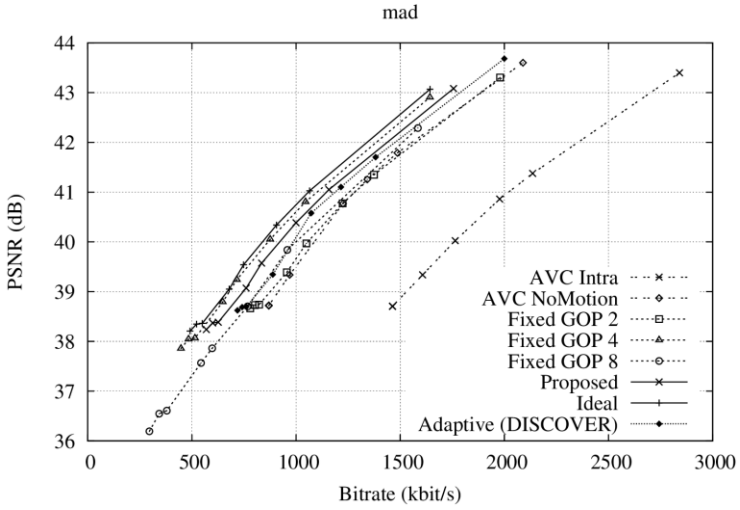
Since many curves are present in the graphs and their performance is very close, Table V reports the BD-PSNR values for the proposed technique compared to the other curves present in the graphs. Excluding the *foreman* case, the gain with respect to the adaptive DISCOVER ranges

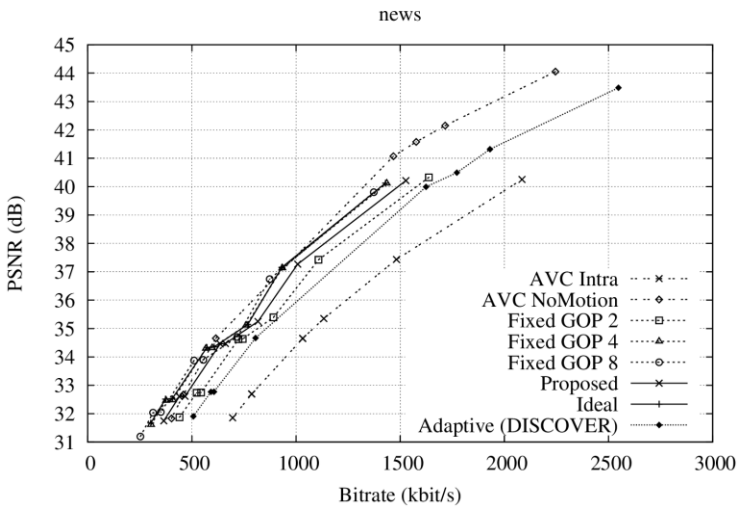
from 0.25 to 2.01 dB PSNR. For some sequences, the performance of a fixed GOP structure may be close to the "ideal" curve. This can be surmised from the table by computing the difference between the column labeled "ideal" and the column of interest. Therefore, in those cases it is difficult to provide gains over a particular fixed structure. Also, in cases such as the *news* sequence, an image with fast motion is shown in the background, hence the classification algorithm may be deceived by the presence of features which are typical of higher-motion conditions, thus GOP size 2 structures are used more than necessary, hence the performance decrease with respect to the case of fixed GOP size 4 and 8. However, we would like to highlight the fact that, even if the performance is not the "ideal" one, the use of a GOP size adaptation algorithm at the encoder is advantageous since it allows to immediately change the size of the GOP when the characteristics of the video content change. An alternative approach could be to perform the GOP size adaptation at the decoder by sending feedback information to the encoder, however this approach presents several issues that would deserve further investigation, i.e., the reaction may not be sufficiently fast in case of rapid changes of the characteristics of the video sequence, and estimating the best GOP size may be difficult due to the use of both past and compressed frames.

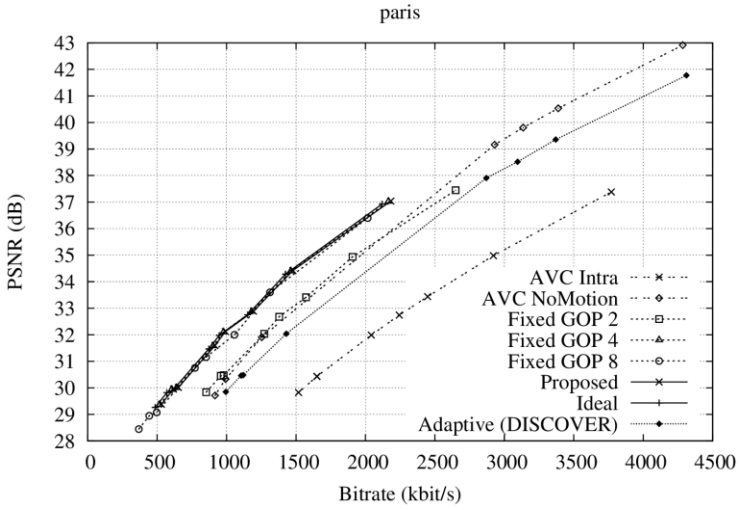
Finally, note that S1, S2 and S4 have been trained using the optimal sequence of GOP sizes as computed for the case of Q=1. Despite this fact, the performance provided by the proposed GOP size selection system does not degrade significantly as Q is increased. This suggests that the optimal GOP size is a characteristic which is strongly dependent on the video content rather than the compression level.

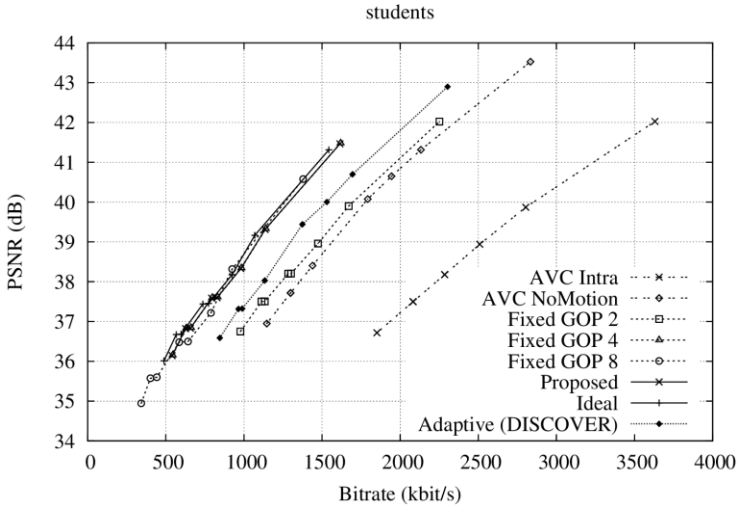

foreman
PSNR (dB)
Bitrate (kbit/s)
AVC Intra
AVC NoMotion
Fixed GOP 2
Fixed GOP 4
Fixed GOP 8
Proposed
Ideal
Adaptive (DISCOVER)


hallmonitor
PSNR (dB)
Bitrate (kbit/s)
AVC Intra
AVC NoMotion
Fixed GOP 2
Fixed GOP 4
Fixed GOP 8
Proposed
Ideal
Adaptive (DISCOVER)


mad
PSNR (dB)
Bitrate (kbit/s)
AVC Intra
AVC NoMotion
Fixed GOP 2
Fixed GOP 4
Fixed GOP 8
Proposed
Ideal
Adaptive (DISCOVER)


news
PSNR (dB)
Bitrate (kbit/s)
AVC Intra
AVC NoMotion
Fixed GOP 2
Fixed GOP 4
Fixed GOP 8
Proposed
Ideal
Adaptive (DISCOVER)


paris
PSNR (dB)
Bitrate (kbit/s)
AVC Intra
AVC NoMotion
Fixed GOP 2
Fixed GOP 4
Fixed GOP 8
Proposed
Ideal
Adaptive (DISCOVER)


students
PSNR (dB)
Bitrate (kbit/s)
AVC Intra
AVC NoMotion
Fixed GOP 2
Fixed GOP 4
Fixed GOP 8
Proposed
Ideal
Adaptive (DISCOVER)

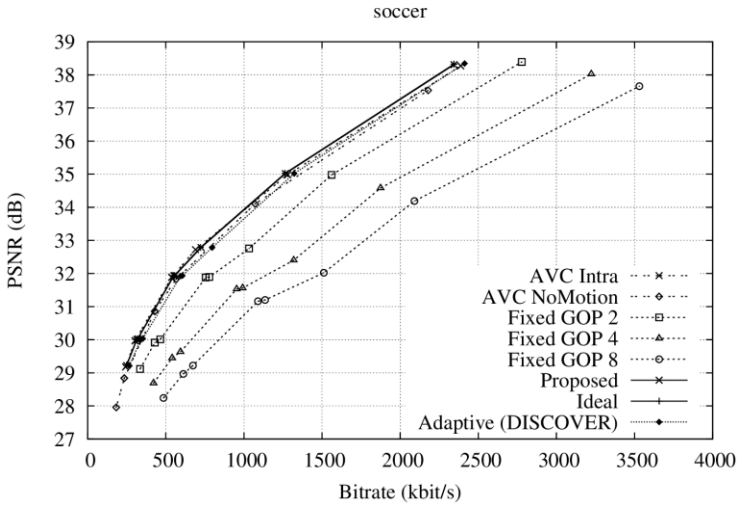


Fig. 12 RD performance for the tested sequences.

TABLE V
PERFORMANCE COMPARISON (BD-PSNR) OF THE PROPOSED TECHNIQUE WITH RESPECT TO THE OTHERS AS DETAILED IN THE TABLE HEADER

Sequence	Ideal	Adaptive DISCOVER	GOP2	GOP4	GOP8	AVC Intra
Foreman	-0.37	-0.07	0.08	0.93	2.14	0.81
Hallmonitor	-0.20	0.83	0.27	-0.19	0.38	2.13
Mad	-0.37	0.25	0.59	-0.23	0.40	3.33
News	-0.39	0.91	0.51	-0.38	-0.39	2.60
Paris	-0.11	2.01	1.30	-0.05	0.11	4.61
Students	-0.15	1.07	1.72	0.00	0.03	5.69
Soccer	-0.02	0.25	1.16	2.47	3.40	0.00

6. Conclusion

This paper presented a framework to optimize the quality of a Wyner-Ziv video encoder by providing a mechanism to adapt the GOP size during the encoding process as a function of the characteristics of the video signal. First, the work showed how to determine the sequence of GOP sizes with the best rate-distortion tradeoff for a DVC encoded video sequence by solving an optimization problem represented by the finding the minimum cost path on a trellis. Then, the best sequences of GOP sizes computed over a set of videos have been used to train three linear support vector machines that can select, at the encoder side, the GOP size whenever a new decision about the GOP size can be taken. The SVM input is a set of features extracted from the original video, considering the current frame and up to four past encoded frames. Results on a set of test video sequences show that the proposed approach outperforms a reference GOP size selection algorithm, and its performance is close to the ideal one. Future work will be devoted to investigating the possibility to implement a similar technique at the decoder and use the feedback channel to convey the information at the encoder as well as to experiment with different types of

features that may improve the SVM classification performance.

References

[1] D. Slepian and J. K. Wolf, "Noiseless coding of correlated information sources," IEEE Transactions on Information Theory , vol.19, no. 4, pp. 471–480, Jul. 1973.

[2] A. D. Wyner and J. Ziv, "The rate-distortion function for source coding with side information at the decoder," IEEE Transactions on Information Theory , vol. 22, no. 1, pp. 1–10, Jan. 1976.

[3] R. Puri and K. Ramchandran, "PRISM: a new robust video coding architecture based on distributed compression principles," in Proceedings of Allerton Conference on Communication,Control and Computing, Allerton, IL, USA, Oct. 2002.

[4] R. Puri, A. Majumdar, and K. Ramchandran, "PRISM: a video coding paradigm with motion estimation at the decoder," IEEE Transactions on Image Processing , vol. 16, no. 10, pp. 2436–2448, Oct. 2007.

[5] A. Aaron, R. U. I. Zhang, and B. Girod, "Wyner-Ziv coding of motion video," in Proceedings of the 36th Asilomar Conference on Signals Systems and Computers, pp. 240–244, Pacific Grove, CA, USA, Nov. 2002.

[6] B. Girod, A. Aaron, S. Rane, and D. Rebollo-Monedero,"Distributed video coding," Proceedings of the IEEE, vol. 93, no. 1, pp. 71–83, Jan. 2005.

[7] A. Aaron, S. Rane, E. Setton and B. Girod, "Transform-Domain Wyner-Ziv Codec for Video", Proc. Of SPIE 5308, VCIP, San Jose, CA, USA, Jan. 2004.

[8] A.Y. Lan, A.G. Nguyen, J.-N. Hwang, "Scene-context-dependent reference-frame placement for MPEG video coding," IEEE Trans. on Circuits and Systems for Video Technology, vol. 9, no. 3, pp. 478-489, Apr. 1999.

[9] J. Ascenso, C. Brites, F. Pereira, "Content adaptive Wyner-Ziv video coding driven by motion activity," Proc. IEEE Intl. Conf. on Image Processing (ICIP), Atlanta, GA, pp. 605-608, Oct 2006.

[10] C. Yaacoub, J. Farah, B. Pesquet-Popescu, "Content adaptive GOP size control with feedback channel suppression in distributed video coding," Proc. IEEE Intl. Conf. on Image Processing (ICIP), Cairo, Egypt, pp. 1397-1400, Nov 2009.

[11] K.R. Vijayanagar, J. Kim, "Dynamic GOP size control for low-delay distributed video coding", Proc. IEEE Intl. Conf. on Image Processing (ICIP), Brussels, Belgium, pp. 157-160, Sep. 2011.

[12] X. Artigas, J. Ascenso, M. Dalai, "The DISCOVER codec: architecture, techniques and evaluation", Proc. of Picture Coding Picture Coding Symposium, Lisbon, Portugal, 2007.

[13] G. Bjøntegaard, "Calculation of average PSNR differences between RD-curves", ITU-T doc. VCEG-M33, April 2001.

[14] Y. Shoham, A. Gersho, "Efficient bit allocation for an arbitrary set of quantizers," IEEE Trans. on Acoustic, Speech and Signal Processing, vol. 36, no. 9, pp. 1445-1453, Sep 1988.

[15] A. Ortega, K. Ramchandran, "Rate-distortion methods for image and video compression," IEEE Signal Processing Magazine, vol. 15, no. 6, pp. 23-50, Nov. 1998.

[16] I. Steinwart, A.Christmann, "Support Vector Machines," Springer, 2008.

[17] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, C.-J. Lin, "Liblinear: A library for large linear classification," J. Machine Learning Research, vol. 9, pp. 1871–1874, Jun. 2008.

[18] C.-C. Chang, C.-J. Lin, "LIBSVM: A library for support vector machines", ACM Trans. on Intelligent Systems and Technology, vol. 2, no. 3, pp. 1-27, Apr. 2011.

[19] Xiph.org foundation, "Xiph.org video test media (derf's collection)," available at http://media.xiph.org/video/derf/

[20] NTIA, "The consumer digital video library", available at http://www.cdvl.org

[21] "Distributed coding for video services (Discover)", available at http://www.discoverdvc.org

[22] J. Ascenso, C. Brites, F. Pereira, "Improving frame interpolation with spatial motion smoothing for pixel domain distributed video coding," Proc. 5th Eurasip Conf. on Speech, Image Processing, Multimedia Communications and Services, Smolenice, Slovak Republic, Jul. 2005.

[23] C. Brites, J. Ascenso, F. Pereira, "Improving transform domain Wyner-Ziv video coding performance," IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP), Toulouse, France, May 2006.